A Framework for Decoupling Overlay SDN and Computing Virtualization

2023 6th World Conference on Computing and Communication Technologies (WCCCT) | 978-1-6654-6146-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/WCCCT56755.2023.10052205

Baohong Lin
China Telecom Research Institute
China Telecom Co.,Ltd.
Guangzhou, China
linbaoh@chinatelecom.cn

Linze Wu
China Telecom Research Institute
China Telecom Co.,Ltd.
Guangzhou, China
wulinz@chinatelecom.cn

Zhilan Huang
China Telecom Research Institute
China Telecom Co.,Ltd.
Guangzhou, China
huangzhil@chinatelecom.cn

Yi Liu
China Telecom Research Institute
China Telecom Co.,Ltd.
Guangzhou, China
liuy311@chinatelecom.cn

Yangchun Li
China Telecom Research Institute
China Telecom Co.,Ltd.
Guangzhou, China
liyc10@chinatelecom.cn

Yongbin Fan
China Telecom Research Institute
China Telecom Co.,Ltd.
Guangzhou, China
fanyongb@chinatelecom.cn

***Abstract*—When enterprise introduces different vendor overlay SDN (Software defined Networking) and computing virtualization in cloud data center, they will face the problem of interoperability. To solve this problem, this paper proposes a framework for decoupling heterogeneous overlay SDN and computing virtualization. The framework has two key elements, that is, the VM (Virtual Machine) lifecycle event mechanism and drainage mechanism based on VM-Type NVE (Network virtualization Edge). VM life cycle event mechanism realizes the synchronization of VM life cycle events among cloud management platform, SDN and computing virtualization. The drainage mechanism realizes the accurate traffic forwarding between business VMs. Test results show that the framework is suitable for enterprise to build a heterogeneous and compatible cloud infrastructure.**

Keywords—SDN, computing virtualization, decoupling, heterogeneous, interoperability

I. INTRODUCTION

At present, the enterprise cloud data center is usually constructed with professional systems such as enterprise cloud management platform, commercial computing virtualization, commercial SDN [1], and commercial storage. Among them, SDN realizes network virtualization and automation, making the network, together with computing and storage, into a pooled virtual resource for tenants to use [2]. SDN solves the problem that traditional physical data center network cannot meet the needs of massive tenants and massive VMs for network isolation, automation, scale and so on in cloud data center [3]. The mainstream SDN solutions are mainly commercial overlay SDN solutions [4] provided by different manufacturers, such as VMware's NSX, Nuage Networks' VSP, Cisco's ACI, and Huawei's Fusion Network. While these commercial overlay SDNs are single-vendor closed solutions. The introduction of commercial overlay SDN in cloud data center will bring problems of interoperability between vendors, professional systems, devices, virtualizations and etc [5]. Specifically, there are problems in the following five aspects [6]:

- Interoperability between overlay SDN and underlay physical network.
- Interoperability between heterogeneous overlay SDN and computing virtualization.
- Interoperability between heterogeneous overlay SDN and enterprise cloud management platform.
- Interoperability between overlay SDN controller and heterogeneous network devices.
- Interoperability between multi cloud data center heterogeneous overlay SDNs.

This paper focuses on the interoperability between heterogeneous overlay SDN and computing virtualization. The key for the interoperability is the interoperation between SDN NVE and computing virtualization hypervisor [7]. Technical implementations of SDN NVE include vSwitch, TOR switch and intelligent network interface card [8]. For vSwitch NVE in most cloud datacenter integration solution, manufacturers use SDN vSwitch to directly replace the native vSwitch of hypervisor, and the hypervisor needs to be re-developed to be able to integrate SDN vSwitch [9]. Obviously, this solution is feasible if only the SDN vendor is the partner of the computing virtualization vendor, otherwise it cannot be implemented.

To solve the problem above, this paper proposes a decoupling framework based on the VM life cycle event mechanism and drainage mechanism based on VM-Type NVE that is the SDN vSwitch deployed in the VM .Based on this framework, we build a cloud infrastructure including open source cloud management platform OpenStack, commercial overlay SDN and commercial computing virtualization. We carry out basic SDN network functions and NVE forwarding performance tests. The test results show the framework can realize the interoperability between heterogeneous overlay

SDN and computing virtualization and has a good bearing effect on performance-insensitive IT systems.

II. METHODOLOGY

As shown in Fig. 1,we have designed a decoupling framework between overlay SDN and computing virtualization, the core modules involved in this framework and the docking between modules are as follows:

- Hypervisor Manager: Hypervisor manager adds a VM event Subscribe-Notice module, which provides the VM lifecycle event subscription and notification interface for OpenStack .
- OpenStack: OpenStack adds a VM event Subscribe-Receive module, which subscribes the required VM life cycle events from the hypervisor manager, completes the matching of relevant resource IDs for the events notified by the hypervisor manager, and then distributes the events to the Neutron module.
- SDN Controller Plugin: The VM event handler module is added to accept the VM life cycle events forwarded by Neutron for processing and conversion to the configuration of SDN controller.
- SDN vSwitch NVE: A special VM which is deployed on each hypervisor node. On one hand, the VM connects to the trunk port group of the hypervisor vSwitch to receive the traffic of other business VMs on the hypervisor node. On the other hand, it communicates with the outside of the hypervisor node by configuring SR-IOV VF or network card pass through. SDN vSwitch NVE is installed inside the VM.

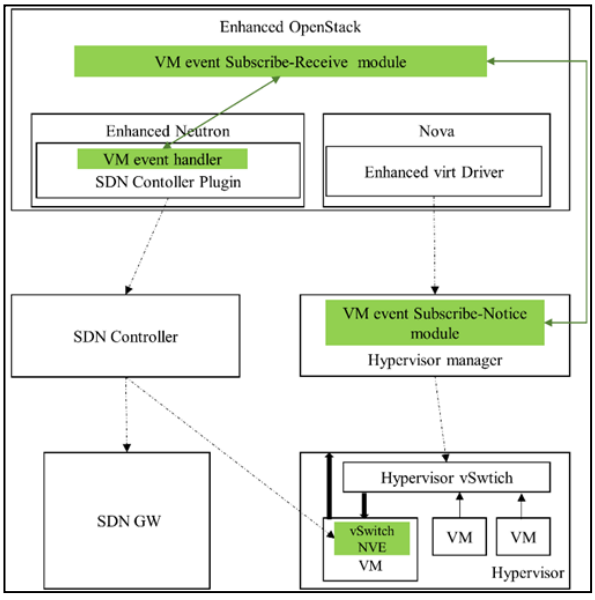

Fig. 1. A framework for decoupling heterogeneous overlay SDN and computing virtualization based on VM-Type NVE and VM lifecycle event mechanism.

We will describe the design of this framework in the following:

A. VM Lifecycle Event Mechanism

1) VM life cycle event type and its value definition

a) Vmpoweron: VM powers on.
b) Livemigratestart: VM starts live migration.
c) Livemigratefinish: VM finishes live migration.
d) Coldmigratestart: VM starts cold migration.
e) Coldmigratestfinish: VM finishes cold migration.
f) Hafinish:high availability for VM is finished.
g) Vmcreated: VM is created.
h) Vmdeleted: VM is deleted.
i) Vmpoweroff: VM powers off.
j) Vifattach: virtual interface is attached.
k) Vifdetach: virtual interface is detached.

2) Interface working mode

OpenStack's VM event Subscribe-Receive module establishes an HTTP long connection with the hypervisor manager's VM event Subscribe-Notice module through http digest authentication, which can be used for event registration / deregistration and event notification.

3) Interface definition

The VM lifecycle event registration /deregistration interface is shown in TABLE I.

TABLE I. VM LIFECYCLE EVENT REGISTRATION / DEREGISTRATION INTERFACE

Name	Type	Description
eventType	String	Registered/deregistered event type, can be vmpoweron/livemigratestart/livemigratefinish/coldmigratestart/coldmigratestfinish/hafinish/vmcreated/vmdeleted/vmpoweroff/vifattach/vifdetach

When a registered event occurs, the hypervisor manager should send an event in a specific format to OpenStack. Interface definition is shown in TABLE II.

TABLE II. HYPERVISOR MANAGER VM LIFECYCLE EVENT NOTIFICATION INTERFACE

Name	Type	Description
event_id	String	Event UUID, independent identification of each event
timestamp	timestamp	Event reporting time
event_type	String	Event reporting type, which can be vmpoweron/livemigratestart/livemigratefinish/coldmigratestart/coldmigratestfinish/hafinish/vmcreated/vmdeleted/vmpoweroff/vifattach/vifdetach
vm_name	String	VM name
vm_uuid	String	VM UUID
vm_status	String	VM running status, the value can be stop / running / paused /, respectively indicating VM stop / VM running / VM pause
host_name	String	Host name of the VM
host_uuid	String	Host UUID of the VM
vnic_list	Array	vNIC (Virtual network interface card) list of VM
vnic_list: vnic_name	String	vNIC name

vnic_list: vnic_uuid	String	vNIC UUID
vnic_list: pg_name	String	Port group name
vnic_list: pg_uuid	String	Port group UUID
vnic_list: mac	String	MAC (Media Access Control) address
vnic_list: port_segmentation_id	Integer	Port group VLAN (Virtual Local Area Network) ID
vnic_list: ip_list	Array	IPv4 address list
vnic_list: ip_list:ip	String	IPv4 address
vnic_list: ipv6_list	Array	IPv6 address list
vnic_list: ipv6_list:ipv6	String	IPv6 address

B. OpenStack Enhancement

1) VM event subscribe-receive module

OpenStack's VM event Subscribe-Receive module interconnects with the hypervisor manager's VM event Subscribe-Notice module through the interface defined in TABLE I. and completes the registration / deregistration of demand events. After receiving the event reported by the hypervisor manager, it complete the matching between the event and the related resource IDs, and then submit the event to the Neutron event handler for processing.

2) Neutron SDN controller plugin event handler

The Neutron SDN controller plugin adds an event handler. After receiving the VM event sented by the OpenStack VM event Subscribe-Receive module, it parses the attributes of the VM event, converts them into the network configurations of business VM, and calls the API (Application Programming Interface) of the SDN controller to send the network configurations.

The interface abstract method of the module is defined as: Public Boolean notify(controller_id, event),its parameters are defined as TABLE III. The event definition is shown in TABLE IV.

TABLE III. PARAMETERS DEFINITION

Name	Type	Description
controllerId	String	Controller UUID, independent identification of the controller
event	JSON	VM lifecycle event

TABLE IV. EVENT DEFINITION

Name	Type	Description
event_id	String	Event UUID, independent identification of each event
timestamp	timestamp	Event reporting time
event_type	String	Reported event type, which can be vmpoweron/livemigratestart/livemigratefinish/coldmigratestart/coldmigratestfinish/hafinish/vmcreated/vmdeleted/vmpoweroff/vifattach/vifdetach
vm_name	String	VM name
vm_uuid	String	VM UUID
vm_status	String	VM running status, the value can be stop / running / paused /, respectively indicating
		VM stop / VM running / VM pause
instance_name	String	OpenStack VM instance name
instance_uuid	String	OpenStack VM instance UUID
host_name	String	Host name of the VM
host_uuid	String	Host UUID of the VM
vnic_list	Array	vNIC list of VM
vnic_list: vnic_name	String	vNIC name
vnic_list: vnic_uuid	String	vNIC UUID
vnic_list: pg_name	String	Port group name
vnic_list: pg_uuid	String	Port group UUID
vnic_list: port_name	String	Port name
vnic_list: port_uuid	String	Port UUID
vnic_list: network_name	String	Network name
vnic_list: network_uuid	String	Network UUID
vnic_list: mac	String	MAC address
vnic_list: port_segmentation_id	Integer	Port group VLAN ID
vnic_list: ip_list	Array	IPv4 address list
vnic_list: ip_list:ip	String	IPv4 address
vnic_list: ipv6_list	Array	IPv6 address list
vnic_list: ipv6_list:ipv6	String	IPv6 address

3) Enhanced neutron port API

To enable Nova virt driver to correctly bind / unbind the vNIC of business VM to the hypervisor vSwitch port group, the extension attribute segmentation_id of the Neutron port API is required to be configured with the VLAN ID of the port group.

4) OpenStack nova enhancement

When SDN vSwitch is deployed in the VM , the network and port created by OpenStack are transferred to SDN controller through Neutron, but the hypervisor manager has no relevant information. In the attach_interface and detach_interface implementation of Nova virt driver, the operation of automatically creating / deleting the port group corresponding to the port is added to enable the hypervisor manager to obtain relevant information.

a) When Nova calls attach_interface to attach the vNIC for the business VM, it determines whether the port group corresponding to the Neutron port exists in the hypervisor node where the VM is located according to the ID and segmentation_id of the Neutron port. If it does not exist, it is automatically created (the naming rule is: [port ID] _ [segmentation_id]) and attached to the VM (One port group corresponds to one port).

b) When Nova calls detach_interface to detach the vNIC for the business VM, it handles the deletion of the corresponding port group.

C. Drainage Scheme of Business VM Traffic

Fig. 2 shows the drainage scheme of this paper, in which:

- vSwitch NVE is deployed in a special VM on the hypervisor. The VM is configured to access the Trunk port group of the hypervisor vSwitch and a SR-IOV VF or a network interface of the physical network interface card. The communication between business VMs is completed through the NVE.
- Each port of the business VM on the hypervisor node accesses a port group with different VLAN tag, which can isolate the communication between VMs. A single node can support up to 4094 VM ports.
- The port of the NVE VM which accesses port group PG_Trunk needs to have layer 2 switching and forwarding capability, that is, pseudo transmission and MAC address change. The pseudo transmission allows the MAC address of the message sent by the NVE VM to be different from the MAC address of the port. The MAC address change allows the NVE VM to receive messages with different MAC addresses of the source MAC and the sending port.
- NVE VM disable automatic migration.

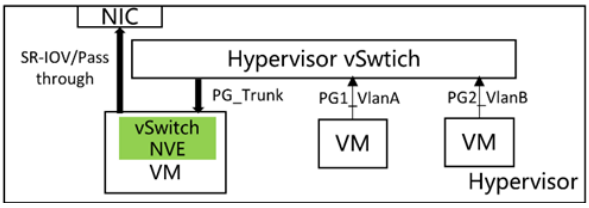

Fig. 2. Drainage scheme of business VM traffic.

III. Test Setup

This paper tests and verify the feasibility of the solution from basic network functions and NVE forwarding performance. The following are the details of the test setup.

A. Hardware Configuration

There are 6 servers (HP DL380 Gen8) to deploy the software system, 2 of which are used as computing nodes to install Hypervisor, 3 as control nodes to install SDN controller, Hypervisor Manager and OpenStack respectively, 1 to install SDN GW(gateway). Each server is equipped with 2-way 6 Core Intel Xeon E5-2640 2.5GHz CPU, 128G DDR4 RAM, 4*2TB HDD, 2 Intel 82599 10G dual-port NIC. The NICs of all servers are connected to 10G switches, and are divided into management plane, control plane and data plane according to networking requirements. Local hard disk space is used for VM storage. SDN NVE forwarding performance test is conducted using Spirent hardware instrument.

B. Software Configuration

We work with computing virtualization and SDN vendors to promote the use of VM lifecycle event mechanism in their products. In this paper, we select a commercial computing virtualization and commercial SDN product, and integrate with our enhanced OpenStack based on our framework.

C. Functional Testing

We carry out functional tests around the test cases shown in TABLE V. that are strongly related to the VM-Type NVE and VM lifecycle event mechanism, such as cloud management platform integration, VM Layer 2 and Layer 3 communication, security group, QOS and etc.

TABLE V. Functional Test Cases

Category	Test Case	Description
cloud management platform integration	Integration of OpenStack, SDN, and computing virtualization to support VM lifecycle event mechanisms	OpenStack, SDN, and computing virtualization can respond correctly when VM lifecycle events occur
L2 network communication	VMs Communication in the same subnet on one NVE	It Supports VMs L2 communication in the same subnet on one NVE
	VMs Communication in the same subnet across NVEs	It Supports VMs L2 communication across NVEs in the same subnet
L3 network communication	VMs Communication across subnets on one NVE	It Supports VMs L3 communication across subnets on one NVE
	VMs Communication across subnets across NVEs	It Supports VMs L3 communication across subnets across NVEs
Security Group	Port Security Group Settings	It Supports adding port to security group and removing port from security group
	Security Group Network Protocol Supported	Security group rules support TCP, UDP, ICMP and other network protocols
QOS	VM's port upstream and downstream bandwidth speed limited	It Supports limiting the upstream and downstream traffic bandwidth of VM's port according to rules and policies, and the error of the rate limiting effect does not exceed 10%
computing virtualization awareness	network properties migration	After VM is migrated, the network attributes (associated ports, subnets, networks and etc) of the VM remain the same as before the migration
	network policies migration	After VM is migrated, the network policies related to the VM (the associated security group, QoS and etc) remain the same as before the migration

D. Performance Testing

We tests the forwarding performance of the NVE to verify how high the performance can be achieved under the physical 10GE link. Spirent Test Center hardware tester is used to test the throughput of NVE according to RFC 2544 [10]. The test configurations are as follows:

- The throughput is tested according to RFC2544. The dataframe length is configured by the Spirent Test Center includes 64 bytes, mixed bytes (about 353 bytes on average), and 1518 bytes for testing.
- The throughput test is configured as dichotomy, bidirectional traffic, the test accuracy is 0.1%, and the judgment threshold is 1‰ frame loss rate allowed.

- Each test is run for 5 rounds, each round is 180 seconds long, the highest and lowest results are removed, and the remaining 3 are averaged as the final result.
- The throughput of four scenarios are tested respectively, including the same subnet on one NVE scenario, across subnets on one NVE scenario, the same subnet across NVEs scenario, and across subnets across NVEs.
- Other parameters are the default configuration of Spirent Test Center.

One NVE scenario is shown in Fig. 3. Two DPDK forwarding VMs and NVE VM are deployed on the same hypervisor, and each DPDK forwarding VM has two vNICs, one vNIC is a pass-through NIC and is directly connected to the 10GE port of the Spirent Test Center, the other vNIC is connected to the Hypervisor vSwitch in the access mode, and the corresponding VLAN is setted on the Hypervisor vSwitch based on intra-subnet or inter-subnet forwarding. L2 unconditional forwarding is set between the two vNICs of the DPDK VM, that is, the traffic flowing in from one vNIC will be forwarded from the other vNIC. The two vNICs of the NVE VM are connected to the Hypervisor vSwitch in trunk mode. Since the communication is on the same host, the VM traffic does not need to be encapsulated as VXLAN(Virtual Extensible Local Area Network) traffic, and is directly forwarded by the NVE through the internal flow table.

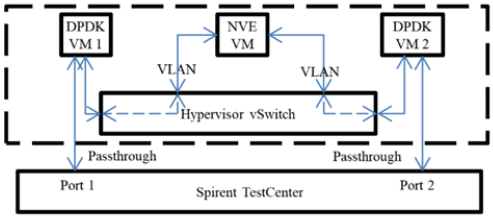

Fig. 3. One NVE performance test scenario.

The cross NVEs scenario is shown in Fig. 4. Different from the one NVE scenario, there are two hypervisor nodes, and each hypervisor has one DPDK forwarding VM and one NVE VM. One vNIC of the NVE VM is a pass through NIC, which is directly connected to the vNIC of NVE on another hypervisor which is also pass through NIC. When the VM traffic is forwarded to the outside of the hypervisor through the NVEs, it will be encapsulated as VXLAN frames.

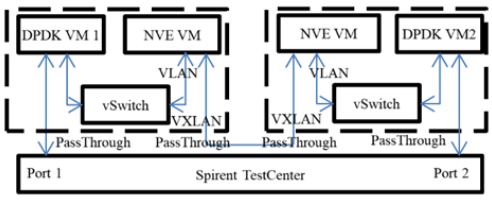

Fig. 4. Cross NVE performance test scenario.

IV. Test Result

The system passes the functional tests, indicating that the heterogeneous Overlay SDN and computing virtualization decoupling and interoperable system based on VM-Type NVE and VM lifecycle event mechanism can support general cloud network functions.

The throughput performance test results of NVE are shown in Fig. 5 and TABLE VI. First of all, in the case of the same frame length, the performance of different scenarios are similar, it can be considered that the forwarding performance of NVE is consistent in different network communication scenarios. The forwarding performance of 64 bytes frames is poor, and bidirectional forwarding rate is below 1Gbps; the performance of 1518 bytes can reach 60%~70% of the line rate, that is, the bidirectional forwarding rate is 6~7Gbps.

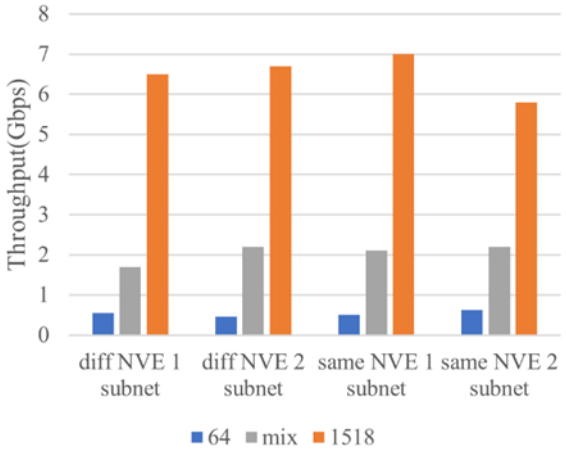

Fig. 5. Throughput test results comparison.

TABLE VI. Throughput Test Results (Gbps)

frame length / scenario	64	1518	mix
different NVE same subnet	0.55	6.5	1.7
different NVE different subnet	0.46	6.7	2.2
same NVE same subnet	0.51	7	2.1
same NVE different subnet	0.62	5.8	2.2

V. Conclusion

This paper proposes a framework to decouple overlay SDN and computing virtualization. The framework provides the ability of standardized interoperability between overlay SDN and computing virtualization through VM lifecycle event mechanism and drainage mechanism based on VM-Type NVE. Testing results of SDN network functions and performance show that the framework can meet the requirements of low network traffic and long average load bytes IT applications in the scenario of heterogeneous overlay SDN and computing virtualization. In the future, we will consider enhancing the capability of the framework to carry performance sensitive services.

REFERENCES

[1] "Software-Defined Networking: The New Norm for Networks," ONF White Paper, April 2012.

[2] S. Azodolmolky, P. Wieder and R. Yahyapour, "SDN-based cloud computing networking," in IEEE International Conference on transparent optical networks, 2013.

[3] F. Callegati, W. Cerroni, C. Contoli and G. Santandrea, "Performance of Network Virtualization in cloud computing infrastructures: The OpenStack case," in IEEE International Conference on Cloud Networking, 2014.

[4] K. Bakshi, "Considerations for Software Defined Networking (SDN): Approaches and use cases," in IEEE International Conference on Aerospace, 2013.

[5] B. Sokappadu, A. Hardin, A. Mungur and S. Armoogum, "Software Defined Networks: Issues and Challenges," in IEEE International Conference on Next Generation Computing Applications, 2019.

[6] O. Tkachova, M. J. Salim and A. R. Yahya, "An analysis of SDN-OpenStack integration," in IEEE International Conference on Scientific-Practical Problems of Infocommunications Science and Technology, 2015.

[7] A. Blenk, A. Basta and W. Kellerer, "HyperFlex: An SDN virtualization architecture with flexible hypervisor function allocation," in IFIP/IEEE International Symposium on Integrated Network Management, 2015.

[8] F. Daniel, et al, "Azure Accelerated Networking:{SmartNICs} in the Public Cloud," in USENIX Symposium on Networked Systems Design and Implementation , 2018.

[9] P. Patel, V. Tiwari and M. K. Abhishek, "SDN and NFV integration in openstack cloud to improve network services and security," in IEEE International Conference on Advanced Communication Control and Computing Technologies, 2016.

[10] B. Scott, and J. McQuaid. "RFC2544: Benchmarking Methodology for Network Interconnect Devices." ,1999.